Relasi Rekurensi

Achmad Ridok dan Tim Pengampu DAA

Bahasan

1. **Relasi Rekurensi**
2. **Metode Penyelesaian Relasi Rekurensi**
3. **Analisa Algoritma Rekursif**

1. Relasi Rekuresnsi

- **Fungsi rekursif :**
  Contoh :

$$T(n)=\begin{cases} c & n=1 \\ 2T\left(\dfrac{n}{2}\right)+cn & n>1 \end{cases}$$

- ***rekurensi*** **: persamaan yang mendeskripsikan suatu fungsi berdasarkan nilainya pada fungsi yang lebih kecil**

$$s(n)=\begin{cases} 0 & n=0 \\ c+s(n-1) & n>0 \end{cases}$$

$$s(n)=\begin{cases} 0 & n=0 \\ n+s(n-1) & n>0 \end{cases}$$

$$T(n)=\begin{cases} c & n=1 \\ 2T\left(\dfrac{n}{2}\right)+c & n>1 \end{cases}$$

$$s(n)=\begin{cases} 0 & n=0 \\ n+s(n-1) & n>0 \end{cases}$$

2. Metode Penyelesaian Relasi Rekurensi

a. **Metode Substitusi**
b. **Metode Iteration**
c. **Metode Master**

T (n) = T + n

a. Metode Substitusi

Dalam metode substitusi ada dua tahap :

1. **Tebak solusi**
2. **Gunakan induksi matematika untuk menemukan syarat batas yang menunjukkan bahwa tebakannya benar**.

a. Metode Substitusi

Contoh 1 : $T(n) = T(n/2) + n$,

- Tebakan : O(logn)
- Untuk konstanta c, tunjukkan :

  $T(n) \leq c \log n$
- $T(n) \leq c \log n$

  $\leq c \log (n/2) + 1$

  $= c \log n - c \log 2 + 1$

  $\leq c \log n$ untuk $c \geq 1$
- Jadi $T(n) = O(\log n)$

Contoh 2 : $T(n) = 2\,T(n/2) + n, n > 1$

- *Tebakan* : $O(n \log n)$
- Untuk konstanta c , tunjukkan :
- $T(n) \leq c\, n \log n$

  $\leq 2\, c\, (n/2) \log (n/2) + n$

  $\leq cn \log n - cn \log 2 + n$

  $= cn \log n – n(c \log 2 - 1)$

  $\leq cn \log n \ (c \geq 1)$
- Jadi $T(n) = 0(n \log n)$

b. Metode iterasi

Contoh 1 :

$$T(n) = \begin{cases} 0 & n = 0 \\ c + T(n-1) & n > 0 \end{cases}$$

Solusi :

T(n) = $\underline{c + T(n-1)}$

= c + c + T(n-2)

= $\underline{2c + T(n-2)}$

= 2c + c + T(n-3)

= $\underline{3c + s(n-3)}$

…

= $\underline{kc + T(n-k)}$

untuk (n-k) = 0, n = k

T(n) = cn + T(0) = cn

Jadi T(n) = O(n)

Contoh 2 :

- $\boldsymbol{T(n) = \begin{cases} 1, \; if\ n = 1 \\ 2T(n-1), if\ n > 1 \end{cases}}$

Solusi :

T(n) = $\underline{2\,T(n-1)}$

= 2[2T(n-2)] = $\underline{2^2T(n-2)}$

= 4[2T(n-3)] = $\underline{2^3T(n-3)}$

. . .

$\underline{= 2^kT(n-k)}$

Untuk n-k=1 maka k = n-1

T(n) = $2^{n-1}T(1) = 2^{n-1}.1 = 2^{n-1}$

Jadi T(n) = O(2^{n-1})

b. Metode iterasi

Contoh 3 .

$$T(n)=\begin{cases} 0 & n=0 \\ n+T(n-1) & n>0 \end{cases}$$

T(n) = <u>n + T(n-1)</u>
= <u>n + (n-1) + T(n-2)</u>
= <u>n + (n-1) + (n-2) + T(n-3)</u>
. . .

$$= \sum_{i=n-k+1}^{n} i \quad + \quad T(n-k)$$

u/ (n-k) = 0, T(n-n) = T(0) = 0

$$T(n) = \sum_{i=1}^{n} i \quad + \quad T(0)$$

Jadi T(n) = O(n^2)

Contoh 4.

$$T(n)=\begin{cases} c & n=1 \\ 2T\left(\frac{n}{2}\right)+c & n>1 \end{cases}$$

T(n) = <u>2T(n/2) + c</u>
= 2(2T(n/2/2) + c) + c
= <u>2^2T(n/2^2) + 2c + c</u>
…
= 2^kT(n/2^k) + ..+$\sum_{i=1}^{k} ic$
Jika (n/2^k) =1 , maka n = 2^k
T(n) = nT(1) + (n-1)c
= n T(1) + (n-1)c
= nc + nc – c = (2n-1)c
Jadi T(n) = O(n)

Latihan 1

Selesai fungsi rekurren berikut dengan metode substitusi dan iterasi :

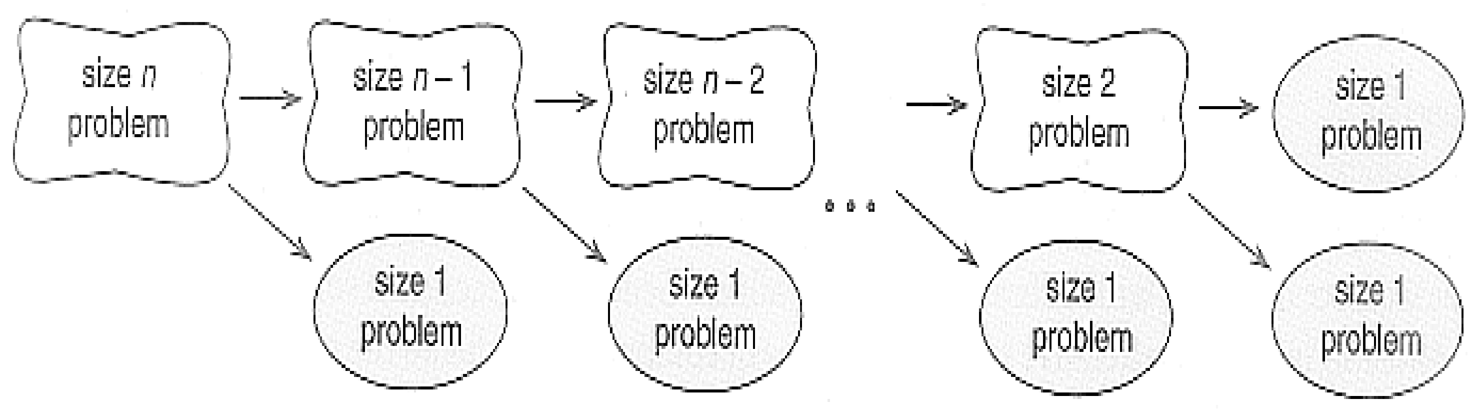
a. **$x(n)=x(n-1)+5$ for $n>1$, $x(1)=0$**
b. **$x(n)=3x(n-1)+5$ for $n>1$, $x(1)=4$**
c. **$x(n)=x(n-1)+n$ for $n>0$, $x(0)=0$**
d. **$x(n)=x(n/2)+n$ for $n>1$, $x(1)=1$ (solve for $n=2^k$)**
e. **$x(n)=x(n/3)+1$ for $n>1$, $x(1)=1$ (solve for $n=3^k$)**

3. Analisa Algoritma Rekursif

Apa itu fungsi rekursif?

- **Fungsi yang memanggil dirinya sendiri**
- **Sebuah fungsi f juga merupakan fungsi rekursif jika memanggil fungsi lain g dan di dalam g terdapat pemanggilan f**

- **Permasalahan yang dapat diselesaikan oleh fungsi rekursif memiliki sifat :**
  - **base case. Contoh 0! = 1.**
  - **recursive cases.**
- **n! = n * (n-1)!**
  - **recursive cases berulang sampai dan sampai pada base case.**
- **n! → (n-1)! → (n-2)! → . . . 1!, 0!.**



Contoh 1. Algoritma Pangkat Rekursif

Algoritma pangkat(X,n) ……………….T (n)
if (n=0) return 1   …………………1
else return X.pangkat(X,n-1)……….T(n-1)+1

- **T**$(n) = \begin{cases} 1 & if\ n = 0 \\ T(n-1) + 1, & if\ n > 0 \end{cases}$

- **Pangkat(5,4)=5*pangkat(5,3)**
  =5*5*pangkat(5,2)
  =5*5*5*pangkat(5,1)
  = 5*5*5*5pangkat(5,0)
  =5*5*5*5*1

Contoh 2 Algoritma Rekursif – Tower Hanoi

Algoritma Tower Hanoi

```
public static void hanoi(int n, char from, char to, char temp){
   if (n == 1)
      System.out.println(from + " -> " + to);
   else{
      hanoi(n - 1, from, temp, to);
      System.out.println(from + " -> " + to);
      hanoi(n - 1, temp, to, from);
   }
}
```

- **The recurrence relation for the running time of the method hanoi is:**
  - **T(n) = a if n = 1**
  - **T(n) = 2T(n - 1) + b if n > 1**

Solusi Alagoritma Towers Hanoi

Expanding:

$T(1) = a, (1)$

$T(n) = 2T(n-1) + b, \text{ if } n > 1$

$= 2^2 T(n-2) + 2b + b$

$= 2^3 T(n-3) + 2^2 b + 2b + b$

$= 2^4 T(n-4) + 2^3 b + 2^2 b + 2^1 b + 2^0 b$

$= 2^k T(n-k) + b[2^{k-1} + 2^{k-2} + \ldots 2^1 + 2^0]$

$$= 2^k T(n-k) + b\sum_{i=0}^{k-1} 2^i$$

$$= 2^k T(n-k) + b(2^k - 1)$$

- **The base case is reached when $n - k = 1 \rightarrow k = n - 1$, we then have:**

$$T(n) = 2^{n-1} T(1) + b(2^{n-1} - 1)$$

$$= (a + b)\, 2^{n-1} - b$$

$$= \left(\frac{a+b}{2}\right) 2^n - b$$

- **Therefore, The method hanoi is $O(2^n)$**

Bentuk-bentuk Algoritma Rekursif

1. $T(n) = T(n-1) + 1$
2. $T(n) = T(n-1) + n$
3. $T(n) = T(n-1) + \log n$
4. $T(n) = T(n/2) + 1$
5. $T(n) = 2T(n-1) + 1$
6. $T(n) = T(n/2) + n$
7. $T(n) = 2T(n/2) + n$

1. Bentuk T(n) = T(n-1) + 1

```
void Alg(int n) { -----> T(n)
    If (n > 0) {
        printf("%d",n) ---> 1
        Alg(n-1); ---------->T(n-1)
    }
}
```

$$T(n) = f(x) = \begin{cases} 1, & n = 0 \\ T(n-1) + 1, & n > 0 \end{cases}$$

- **Solusi Iterasi**

T(n) = T(n-1) + 1 -------> 1

T(n) = [T((n-1)-1) + 1] + 1

T(n) =T(n-2) +2 -------->2

T(n) =[T((n-2)-1) + 1]+ 2

T(n) =T(n-3) + 3 ------->3

...

T(n) = T(n-k) + k

Asumsi n-k = 0 ➔ n = k

T(n) = T(0) + n = 1 + n

θ(n)

2. Bentuk T(n) = T(n-1) + n

```
void Alg(int n) {---------> T(n)
      If(n > 0) {
          for (i = 0; i < n; i++) {
              printf("%d",n) -----> n
          }
          Alg(n-1)  -------------> T(n-1)
}
```

$$T(n) = \begin{cases} 1, & n = 0 \\ T(n-1) + n, & x > 0 \end{cases}$$

- **Solusi Iterasi**

T(n) = T(n-1) + n -----------------> 1
T(n) = [T((n-1)-1) + (n-1)] + n
T(n) = T(n-2) + (n-1) + n --------->2
T(n) = [T(n-2)-1) + (n-2)] + (n-1) + n
T(n) = T(n-3) + (n-2) + (n-1) + n
…
T(n) = T(n-k) + (n-(k-1)) +(n-(k-2) + . . .+ (n – 2) + (n – 1) + n
Asumsi n-k = 0 ➔ n = k
T(n) =T(0) + 1 + 2 + . . .+ (n – 1) + (n – 1) + n
T(n) = 1 + 1 + 2 + . . .+ n = 1 + n(n+1)/2
===> $\theta(n^2)$

3. Bentuk T(n) = T(n-1) + log n

```
void Alg(int n) { -------------> T(n)
  if (n > 0) {
    for (i = 1; i < n; i=i*2) {
      printf(“%d”,i);  -----> log n
    }
    Alg(n-1); -----> T(n-1)
}
```

$$T(n) = \begin{cases} 1, & n = 0 \\ T(n-1) + \log n, & n > 0 \end{cases}$$

- **Solusi Iterasi**

T(n) = T(n-1) + log n -------> 1
T(n) = [T(n-1)-1) + log (n-1)] + log n
T(n) = T(n-2) + log (n-1) + log n ------> 2
….
T(n) = T(n-k) + log (n – (k-1) + . . . + log (n- 1) + log n
Asumsi n-k = 0 ➔ n = k
T(n) = T(0) + log (n – n + 1) + . . .+ log (n-1) + log n
T(n) = 1 + log (1) + log (2) + . . .+ log (n -1) + log n
T(n) = 1 + log (1*2*3*. . .(n-1)*n)
T(n) = 1 + log n!
O(n log n)

4. Bentuk T(n) = T(n/2) + 1

```
Alg(int n) {   -------------------> T(n)
      if( n > 1) {
         printf("%d",n) -----> 1
         Alg (n/2); ---------->T(n/2)
      }
}
```

T(n) = T(n/2) + 1

$$T(n) = \begin{cases} 0, & n = 1 \\ T\left(\frac{n}{2}\right) + 1, & n > 1 \end{cases}$$

- **Solusi Iterasi**

T(n) = T(n/2) + 1 ----------> 1

$T(n) = [T(n/2^2) + 1] + 1$

$T(n) = T(n/2^2) + 2$ -------> 2

$T(n) = T(n/2^3) + 3$ ------> 3

..

$T(n) = T(n/2^k) + k$

Diasumsikan $n/2^k = 1 \Rightarrow n = 2^k$ dan $k = \log n$

T(n) = T(1) + log n = 1 + log n

=== > O(log n)

5. Bentuk T(n) = 2T(n-1) + 1

```
Alg (int n) {   ----- > T(n)
    If (n > 0) {
      print("%d",n); -----> 1
      Alg (n-1); --- > T(n-1)
      Alg (n-1); ---> T(n-1)
    }
}
```

$$T(n) = \begin{cases} 1, & n = 0 \\ 2T(n-1) + 1, & x > 0 \end{cases}$$

- **Solusi Iteasi**

T(n) = 2 T(n-1) + 1 ----------------> 1

$T(n) = 2[2T((n-1)-1) + 1] + 1$

$T(n) = 2^2T(n-2) + 2 + 1$ ----------> 2

$T(n) =2^2[2T((n-2)-1) + 1] + 2 + 1$

$T(n) = 2^3T(n-3) + 22 + 2 + 1$

. . .

$T(n) = 2^kT(n-k) + 2^{(k-1)} +. . .+ 2^2 + 2^1 + 2^0$

Asumsi n – k = 0 ➔ n = k

$T(n) = 2^nT(0) + 2^{n-1} + . . .+ 2^2 + 2^1 + 2^0$

$T(n) = 2^n + 2^{n-1} + . . .+ 2^2 + 2^1 + 2^0$

=== > $O(2^n)$

6. Bentuk T(n) = T(n/2) + n

$$T(n) = f(x) = \begin{cases} 1, & n = 1 \\ T\left(\frac{n}{2}\right) + n, & n > 1 \end{cases}$$

- **Solusi Iterasi**

$T(n) = T(n/2) + n$ -------------> 1
$T(n) = [T(n/2^2) + n/2] + n$
$T(n) = T(n/2^2) + n/2 + n$ ----->2
$T(n) = [T(n/2^3] + n/2^2] + n/2 + n$
$T(n) = T(n/2^3) + n/2^2 + n/2 + n$ -------> 3
..
$T(n) = T(n/2^k) + n/2^{k-1} + n/2^{k-2} + \ldots + n/2^2 + n/2^1 + n/2^0$
Asumsikan $n/2^k = 1$ ➔ $n = 2^k$ dan $k = \log n$
$T(n) = T(1) + n[1/2^{k-1} + 1/2^{k-2} + . . . + ½ + 1)$
$T(n) = 1 + n(1 + 1) = 1 + 2n$
$O(n)$

7. Bentuk T(n) = 2T(n/2) + n

```
void Alg(int n) {  -------------> T(n)
    if(n > 1) {
        for (i=0; i < n; i++) {
           stms;     ---------------> n
        }
        Alg(n/2)   ---------------> T(n/2)
        Alg(n/2) --------------->T(n/2)
    }
}
```

T(n) = 2T(n/2) + n

$$T(n) = \begin{cases} 1, & n = 1 \\ 2T\left(\frac{n}{2}\right) + n, & n > 0 \end{cases}$$

- **Solusi Iterasi**

$T(n) = 2\,T(n/2) + n$ -----> 1

$T(n) = 2[2T(n/2^2) + n/2] + n$

$T(n) = 2^2T(n/2^2) + n + n$ --------> 2

$T(n) = 2^2[2T(n/2^3] + n/2^2] + n + n$

$T(n) = 2^3T(n/2^3) + n + n + n$ ----->3

....

$T(n) = 2^kT(n/2^k) + kn$

Asumsi $n/2^k = 1 \rightarrow n = 2^k$ dan $k = \log n$

$T(n) = nT(1) + n \log n = n + n \log n$

=== > $O(n \log n)$

Latihan 2

Tuliskan relasi rekurensi berikut dan tentukan kelas OoG-nya

1.

```
A (n)
   if (n=1) return 2
   else return 3 * A(n/2) + A(n/2) + 5
```

2.

```
 A(n)
     if (n =1 or n = 2) return 1
     else return A(n-1) + A(n-2)
```

3.

```
Rek3(n)
      if (n > 0) then
       do something
       Rek3(n-1)
       Rek3(n-1)
   endif
```

Terima Kasih